

WHAT IS CLAIMED IS:

- 1 1. An application programming interface comprising:
2 a first interface which controls transfer of information between a first
3 device capable of handling isochronous and asynchronous data and an audio/video file
4 system capable of handling and organizing audio/video data.
- 1 2. The application programming interface according to claim 1,
2 further comprising:
3 a second interface which controls transfer of information between a second
4 device capable of handling asynchronous data and said audio/video file system.
- 1 3. The application programming interface according to claim 1,
2 wherein said first device is an audio/video controller.
- 1 4. The application programming interface according to claim 3,
2 wherein said audio/video controller is capable of processing commands transmitted using
3 protocol 61883.
- 1 5. The application programming interface according to claim 4,
2 wherein said commands are transmitted using protocol 61883 in an isochronous manner.
- 1 6. The application programming interface according to claim 2,
2 wherein said second device is a SBP-2 controller.
- 1 7. The application programming interface according to claim 6,
2 wherein said SBP-2 controller is capable of processing commands transmitted using
3 serial-bus-protocol-2.
- 1 8. The application programming interface according to claim 7,
2 wherein said commands are transmitted using serial-bus-protocol-2 in an asynchronous
3 manner.
- 1 9. The application programming interface according to claim 1,
2 wherein control of said transfer of information to and from said first device are
3 independent of internal implementation of said first device.

1 10. The application programming interface according to claim 2,
2 wherein control of said transfer of information to and from said second device are
3 independent of internal implementation of said second device.

1 11. The application programming interface according to claim 1 further
2 comprising:
3 a plurality of function calls.

1 12. The application programming interface according to claim 11,
2 wherein one or more of said plurality of function calls are designed to allow said
3 audio/video file system to handle a first type of file; and wherein one or more of said
4 plurality of function calls are designed to allow said audio/video file system to handle a
5 second type of file.

1 13. The application programming interface according to claim 12,
2 wherein said first type of file is a non-audio/video file; and wherein said second type of
3 file is an audio/video file.

1 14. The application programming interface according to claim 12,
2 wherein said first type of file is smaller than said second type of file.

1 15. The application programming interface according to claim 11,
2 wherein one or more of said plurality of function calls are designed to allow said
3 audio/video file system to play or record a plurality of audio/video data streams
4 concurrently.

1 16. The application programming interface according to claim 15,
2 wherein said one or more of said plurality of function calls are designed to allow said
3 audio/video file system to play or record said plurality of audio/video data streams
4 concurrently by using a channel ID parameter and an object ID parameter.

1 17. The application programming interface according to claim 11,
2 wherein one or more of said plurality of function calls are designed to allow said
3 audio/video file system to play and record an audio/video data stream concurrently.

1 18. The application programming interface according to claim 17,
2 wherein said one or more of said plurality of function calls are designed to allow said
3 audio/video file system to play and record said audio/video data stream concurrently by
4 using a channel ID parameter and an object ID parameter.

1 19. The application programming interface according to claim 11,
2 wherein one or more of said plurality of function calls are designed to allow said
3 audio/video file system to initiate a play or record operation starting from within an
4 audio/video file.

1 20. The application programming interface according to claim 19,
2 wherein said one or more of said plurality of function calls are designed to allow said
3 audio/video file system to initiate a play or record operation starting from within said
4 audio/video file by using an offset parameter.

1 21. The application programming interface according to claim 11,
2 wherein one or more said plurality of function calls are designed to allow said
3 audio/video file system to optimize disk access.

1 22. The application programming interface according to claim 21,
2 wherein said one or more of said plurality of function calls are designed to allow said
3 audio/video file system to optimize disk access by designating a first group of function
4 calls to handle a first type of file and a second group of function calls to handle a second
5 type of file.

1 23. The application programming interface according to claim 22,
2 wherein said first type of file is a non-audio/video file; and wherein said second type of
3 file is an audio/video file.

1 24. The application programming interface according to claim 11,
2 wherein one or more of said plurality of function calls are designed to allow said
3 audio/video file system to perform a plurality of trick operations with a data stream.

1 25. The application programming interface according to claim 24,
2 wherein said plurality of trick operations includes a plurality of forward operations.

1 26. The application programming interface according to claim 25,
2 wherein said plurality of forward operations includes a fast-forward operation, a slow-
3 forward operation, and a step-forward operation.

1 27. The application programming interface according to claim 24,
2 wherein said plurality of trick operations includes a plurality of reverse operations.

1 28. The application programming interface according to claim 27,
2 wherein said plurality of reverse operations includes a fast-reverse operation, a slow-
3 reverse operation, and a step-reverse operation.

1 29. An application programming interface for providing an interface
2 with an audio/video file system capable of handling and organizing audio/video data,
3 comprising:
4 a first plurality of function calls including:
5 a load function call designed to cause retrieval of descriptor
6 information from a storage medium;
7 a store function call designed to cause storing of said descriptor
8 information onto said storage medium;
9 a delete function call designed to cause deletion of said descriptor
10 information from said storage medium; and
11 a second plurality of function calls including:
12 a play function call designed to cause a specified file to be played;
13 a record function call designed to cause specified data to be
14 recorded; and
15 a stop function call designed to cause a play or record operation to
16 be stopped.

1 30. The application programming interface according to claim 29,
2 wherein said first plurality of function calls is designed to handle a first type of file; and
3 wherein said second plurality of function calls is designed to handle a second type of file.

1 31. The application programming interface according to claim 30,
2 wherein said first type of file is a non-audio/video file; and wherein said second type of
3 file is an audio/video file.

1 32. The application programming interface according to claim 29,
2 wherein said first plurality of function calls further includes:
3 a validity function call designed to verify validity of a specified
4 descriptor; and
5 wherein said second plurality of function calls further includes:
6 a pause function call designed to cause a play or record operation
7 to be paused;
8 a resume function call designed to cause a previously paused
9 operation to resume; and
10 an address retrieval function call designed to determine a logical
11 block address of said specified file during a play or a record operation.

1 33. The application programming interface according to claim 29,
2 wherein said second plurality of function calls includes:
3 a plurality of function calls designed to cause forward operations to be
4 performed; and
5 a plurality of function calls designed to cause reverse operations to be
6 performed.

1 34. The application programming interface according to claim 33,
2 wherein said plurality of function calls designed to cause forward operations to be
3 performed includes:
4 a fast-forward function call;
5 a slow-forward function call; and
6 a step-forward function call.

1 35. The application programming interface according to claim 33,
2 wherein said plurality of function calls designed to cause reverse operations to be
3 performed includes:
4 a fast-reverse function call;
5 a slow-reverse function call; and
6 a step-reverse function call.

1 36. The application programming interface according to claim 29,
2 wherein said application programming interface is capable of being used by a first device

3 capable of handling isochronous and asynchronous data to communicate with said
4 audio/video file system.

1 37. The application programming interface according to claim 36,
2 wherein said first device is an audio/video controller.

1 38. The application programming interface according to claim 36,
2 wherein said application programming interface is capable of being used by a second
3 device capable of handling asynchronous data to communicate with said audio/video file
4 system.

1 39. The application programming interface according to claim 38;
2 wherein said first device is a SBP-2 controller.

1 40. The application programming interface according to claim 32,
2 wherein said specified descriptor is an object descriptor.

1 41. The application programming interface according to claim 32,
2 wherein said specified descriptor is a content list.

1 42. The application programming interface according to claim 32,
2 wherein said specified descriptor is a performance list.

1 43. The application programming interface according to claim 32,
2 wherein said specified descriptor is a HMS table.

1 44. The application programming interface according to claim 32,
2 wherein each of said first and second plurality of function calls is capable of passing a
3 plurality of parameters.

1 45. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said load function
3 call includes a descriptor ID parameter, a type parameter, an offset parameter, a size
4 parameter, a data_location parameter, and a call_back parameter.

1 46. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said store function

3 call includes a descriptor ID parameter, a type parameter, an offset parameter, a size
4 parameter, a data_location parameter, and a call_back parameter.

1 47. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said delete
3 function call includes a descriptor ID parameter, a type parameter, and a call_back
4 parameter.

1 48. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said play function
3 call includes a channel ID parameter, an object ID parameter, a start_position parameter,
4 an_end position parameter, a speed parameter, and a call_back parameter.

1 49. The application programming interface according to claim 44,
2 wherein said plurality of parameters that are capable of being passed by said record
3 function call include a channel ID parameter, an object ID parameter, a start_position
4 parameter, a type parameter, and a call_back parameter.

1 50. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said stop function
3 call includes a channel ID parameter, a call_back parameter, and a logical_byte_address
4 parameter.

1 51. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said pause function
3 call includes a channel ID parameter, a call_back parameter, and a logical_byte_address
4 parameter.

1 52. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said resume
3 function call includes a channel ID parameter and a call_back parameter.

1 53. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said address
3 retrieval function call includes a channel ID parameter and a count parameter.

1 54. The application programming interface according to claim 44,
2 wherein said plurality of parameters that is capable of being passed by said validity
3 function call includes a descriptor ID parameter, a type parameter and a call_back
4 parameter.

1 55. The application programming interface according to claim 34,
2 wherein said fast-forward function call is capable of passing a plurality of parameters
3 including a channel ID parameter, a type parameter, an interval parameter, a repeat
4 parameter, and a call_back parameter.

1 56. The application programming interface according to claim 34,
2 wherein said slow-forward function call is capable of passing a plurality of parameters
3 including a channel ID parameter, a repeat parameter, an increment parameter and a
4 call_back parameter.

1 57. The application programming interface according to claim 34,
2 wherein said step-forward function call is capable of passing a plurality of parameters
3 including a channel ID parameter, an increment parameter and a call_back parameter.

1 58. The application programming interface according to claim 35,
2 wherein said fast-reverse function call is capable of passing a plurality of parameters
3 including a channel ID parameter, a type parameter, an interval parameter, a repeat
4 parameter, and a call_back parameter.

1 59. The application programming interface according to claim 35,
2 wherein said slow-reverse function call is capable of passing a plurality of parameters
3 including a channel ID parameter, a repeat parameter, an increment parameter and a
4 call_back parameter.

1 60. The application programming interface according to claim 35,
2 wherein said step-reverse function call is capable of passing a plurality of parameters
3 including a channel ID parameter, an increment parameter and a call_back parameter.

1 61. A method for providing communication with an audio/video file
2 system, comprising steps of:

3 providing a first interface which controls transfers of information between
4 said audio/video system and a first device capable of handling isochronous and
5 asynchronous data; and
6 providing a second interface which controls transfers of information
7 between said audio/video system and a second device capable of handling asynchronous
8 data.

1 62. The method according to claim 61, wherein said signals transferred
2 between said audio/video system and said first device are independent of internal
3 implementation of said first device; and
4 wherein said signals transferred between said audio/video system and said
5 second device are independent of internal implementation of said second device.

10005884-120301
T0E02T-4885000T